

Generalised k -Steiner Tree Problems in Normed Planes

Marcus Brazil, Charl J. Ras, Konrad J. Swanepoel, Doreen A. Thomas

Abstract

The 1-Steiner tree problem, the problem of constructing a Steiner minimum tree containing at most one Steiner point, has been solved in the Euclidean plane by Georgakopoulos and Papadimitriou using plane subdivisions called oriented Dirichlet cell partitions. Their algorithm produces an optimal solution within $\mathcal{O}(n^2)$ time. In this paper we generalise their approach in order to solve the k -Steiner tree problem, in which the Steiner minimum tree may contain up to k Steiner points for a given constant k . We also extend their approach further to encompass arbitrary normed planes, and to solve a much wider class of problems, including the k -bottleneck Steiner tree problem and other generalised k -Steiner tree problems. We show that, for any fixed k , such problems can be solved in $\mathcal{O}(n^{2k})$ time.

Keywords: k -Steiner tree; bottleneck Steiner problem; network optimisation; polynomial-time algorithm

1 Introduction

The *Steiner tree problem*, which asks for a network with minimum total edge length interconnecting a given set of points (called *terminals*), is a well known geometric variant of the *spanning tree problem*. The Steiner tree problem can be viewed as belonging to a family of problems where the aim is to construct a network T interconnecting a given set of n terminals, with the following properties:

1. T may contain nodes other than the given terminals;
2. T minimises a given cost function;
3. the given cost function guarantees that T can be assumed to be a minimum spanning tree on its nodes (for a given metric).

In addition to the Steiner tree problem under various metrics, this family also includes the *power- p Steiner tree problem*, where the cost of each edge of the network is the p th power of its length, and the *bottleneck Steiner tree problem*, where the cost of the network is the length of its longest edge and there is a bound on the number of extra nodes in the network. Both of these variants on the Steiner tree problem have numerous applications, particularly in the design of wireless communications networks.

Although the spanning tree problem is polynomially solvable, being solvable in $O(n^2)$ time in a general metric and in $O(n \log n)$ time in the Euclidean plane, see [20], the problems in this more general family are mostly *NP*-complete, even in the plane. This essentially stems from the fact that as the number of extra nodes that can be added to the network increases there is an exponential explosion in the number of different topologies that need to be considered. A natural way of controlling this increase in complexity is to bound the number of extra nodes, in other words, replace Property 1 above by the following:

- 1a. T may contain up to k nodes other than the given terminals, where k is a given positive integer.

We refer to problems in this modified class as *generalised k -Steiner tree problems*. Under suitable conditions, such problems are polynomially solvable. Understanding the efficiency with which such problems can be solved, when terminals are embedded in a normed plane, is the main topic of this paper.

One of the seminal papers on this topic, for the 1-Steiner tree problem in the Euclidean plane, is a paper by Georgakopoulos and Papadimitriou [6], where an $O(n^2)$ -time solution is given. They conclude their paper with a tantalising comment relating to their unsuccessful attempts at generalising their methodology to the k -Steiner tree problem, even for $k = 2$. This comment has been a motivating factor for the current paper. Also, in their paper the proofs of some of the primary results are omitted, and some lack sufficient detail; another aim of this paper is to add some rigour to the more fundamental of these results.

We may restate the goal of the Georgakopoulos and Papadimitriou paper as follows: find the point in the Euclidean plane which, if added to a given set of points, will result in the shortest possible spanning tree. The authors observed that one can significantly reduce the time-complexity of an algorithmic solution to the problem by first constructing a special partition. Given a set X of n terminals in the Euclidean plane, it is possible to partition the plane into $O(n^2)$ regions such that if any new point \mathbf{s} is embedded in the plane within one of these regions, say R , then any minimum spanning tree T on $X \cup \{\mathbf{s}\}$ will have the following property: the neighbours of \mathbf{s} in T will belong to some subset of a set $C_X(R)$ containing at most six points from X , where $C_X(R)$ is fixed for the given region R . This useful partition is referred to as the *overlayed oriented Dirichlet cell partition*, or OODC partition. Their algorithm takes as input the set X of terminals and then starts by calculating the OODC partition, the set $C_X(R)$ for every R , and a minimum spanning tree T' on X . All this, as well as a preprocessing step on T' , is done within a time of $O(n^2)$. For each region R , the algorithm then iterates through all subsets S of $C_X(R)$ and calculates \mathbf{s} , the Steiner point of the nodes in S (this takes constant time for each S). The algorithm then updates T' (which can also be done in constant time because of the preprocessing step) to include \mathbf{s} . The cheapest tree is selected at the end as an optimal solution. A naive algorithm for the 1-Steiner tree problem would attain a complexity of $O(n^6 \cdot n \log n)$ since it would have to iterate through *all* subsets of up to six terminals and then calculate the optimal position of the Steiner point and the corresponding minimum spanning tree for each of these subsets.

The powerful simplifying properties of the OODC would clearly be advantageous in a generalisation of the algorithm to arbitrary normed planes. However, the construction of the OODC partition given in [6] is valid for the Euclidean plane only. We will provide a new method, based on abstract Voronoi diagrams, for constructing the partition for terminals embedded in an arbitrary normed plane. The construction is based on theoretical results, but we also define a class of norms for which the algorithm would be practically implementable. Once the partition has been found, our algorithm calculates the optimal positions of all k Steiner points simultaneously. Of course, these positions will depend not only on the neighbours of the Steiner points, but also on the cost function of the given generalised Steiner tree problem. Since, at the start of this step, the neighbours of the Steiner points are fixed but the Steiner points are free, this subproblem is a generalised version of the well-known *fixed topology Steiner tree problem* (discussed in Section 4). Even though we consider k to be part of the input of our algorithm, k will generally be much smaller than n , and therefore we assume that this step can be done in constant time. A novel method for updating a minimum spanning tree is then utilised to calculate a potential solution for every choice of coordinates of the Steiner points. Once again, a cheapest tree is selected as the optimal solution. The total time-complexity turns out to be $\mathcal{O}(n^{2k})$ when factors that are not functions of n are excluded.

There are a number of authors who have looked at adapting the solution to the 1-Steiner tree problem in [6] to other ℓ_p norms. Kahng and Robins in [12] do this for the rectilinear plane, however, their paper only uses the solution as a step in a heuristic algorithm for the rectilinear Steiner tree problem, and not much attention is devoted to the solution of the 1-Steiner tree problem itself. Griffith et al. expand on this heuristic idea in the rectilinear plane, see [8], and they provide a simple procedure (though without proof) for updating a minimum spanning tree when a new node is introduced. Lin et al. in turn adapt the approach presented by Kahn and Robins to the λ_3 -plane in [15].

Section 2 provides some preliminary definitions. We define the OODC partition in Section 3 and provide details, including the time-complexity, of its construction. We also present three restrictions on the given normed plane that would allow the construction to be implemented in practice. The problem of locating Steiner points under a fixed topology is outlined in Section 4. Many results on this topic exist in the literature, so we summarise them very briefly for a few of the more common instances of the generalised Steiner tree problem. Section 5 describes a new method, and furnishes a correctness proof, of updating a minimum spanning tree to contain a fixed subtree. In Section 6 we present our algorithm for calculating an optimal solution to any given instance of a generalised k -Steiner tree problem in a normed plane, and then prove its correctness and verify its time-complexity.

2 Preliminaries

We begin by formalising the definition of a generalised k -Steiner tree problem, sketched in the introduction.

Throughout this paper we use the symbols $E(G)$ and $V(G)$ for the edge-set and node-set

respectively of a graph G . We also use the notation $G = \langle V(G), E(G) \rangle$. Let $k' > 0$ be given; let $\|\cdot\|$ be a given norm on \mathbb{R}^2 .

Given a set $P = \{p_1, \dots, p_{n+k'}\}$, let $\{\mathcal{T}\}$ represent the set of all spanning trees for the elements of P . For each \mathcal{T} there is a corresponding set of edges $E(\mathcal{T}) = \{e_1, \dots, e_{n+k'-1}\}$ (with each $e_i \in P \times P$). Let $X = \{x_1, \dots, x_n\}$, with each $x_i \in \mathbb{R}^2$, represent an embedding of the set $\{p_1, \dots, p_n\}$ in \mathbb{R}^2 (where each x_i is an embedding of the corresponding p_i). We can think of each \mathcal{T} as representing the topology of a tree network interconnecting X and using k' extra nodes, and we can equate the edges $E(\mathcal{T})$ with the arcs of such a network. For a fixed embedding of this network we let $S = \{x_{n+1}, \dots, x_{n+k'}\}$, with each $x_i \in \mathbb{R}^2$, be the locations of the extra nodes corresponding to $\{p_{n+1}, \dots, p_{n+k'}\}$. We refer to X as the set of *terminals* and S as the set of *Steiner points* of the network. Now let $\mathbf{e}_{\mathcal{T}, X, S} = (\|e_1\|, \dots, \|e_{n+k'-1}\|)$; i.e., the components of $\mathbf{e}_{\mathcal{T}, X, S}$ are the edge lengths of such a network, for a given tree topology and a given set of embedded nodes. Such a vector is well defined up to the order of its components.

Let $\alpha : \mathbb{R}_+^{n+k'-1} \rightarrow \mathbb{R}$ be a symmetric function (ie, independent of the order of the components of the vector on which it acts). We think of α as a cost function on a tree network. In other words, $\alpha(\mathbf{e}_{\mathcal{T}, X, S})$ is the cost of the network with topology \mathcal{T} and nodes X and S , and $\min_{\mathcal{T}, S} \alpha(\mathbf{e}_{\mathcal{T}, X, S})$ is the minimum cost (with respect to α) of any tree interconnecting the nodes X and k' other points. Hence, for the power- p Steiner tree problem we define

$$\alpha(\mathbf{e}_{\mathcal{T}, X, S}) = \alpha_p(\mathbf{e}_{\mathcal{T}, X, S}) := \sum_1^{n+k'-1} \|e_i\|^p;$$

whereas, for the bottleneck problem (where the cost of the network is the cost of the longest edge) we have

$$\alpha(\mathbf{e}_{\mathcal{T}, X, S}) = \alpha_\infty(\mathbf{e}_{\mathcal{T}, X, S}) = \max_{E(\mathcal{T})} \|e_i\|.$$

We define such a symmetric function α to be ℓ_1 -*optimisable* if and only if there exist \mathcal{T}^* and S^* such that $\alpha(\mathbf{e}_{\mathcal{T}^*, X, S^*}) = \min_{\mathcal{T}, S} \alpha(\mathbf{e}_{\mathcal{T}, X, S})$ and $\alpha_1(\mathbf{e}_{\mathcal{T}^*, X, S^*}) = \min_{\mathcal{T}} \alpha_1(\mathbf{e}_{\mathcal{T}, X, S^*})$. In other words, α is ℓ_1 -*optimisable* if for any given X there exists a tree T interconnecting X , with minimum cost with respect to α , that is a minimum spanning tree on its *complete set* of nodes. We denote the cost of such a tree by $\|T\|_\alpha$. It is easy to show that α_p , for $p > 0$, and α_∞ are ℓ_1 -optimisable.

Definition. For any given positive integer k , a *generalised k -Steiner tree problem* is defined to be any problem of the following form:

Given A set X of n points in \mathbb{R}^2 , a norm $\|\cdot\|$, and a symmetric ℓ_1 -optimisable function α .

Find A set S of $k' \leq k$ points in \mathbb{R}^2 , and a spanning tree T on $X \cup S$ with topology \mathcal{T} such that $\|T\|_\alpha = \alpha(\mathbf{e}_{\mathcal{T}, X, S}) = \min_{\mathcal{T}', S'} \alpha(\mathbf{e}_{\mathcal{T}', X, S'})$.

We refer to T as a *generalised k -Steiner minimum tree*. The next lemma is an extension of the Swapping Algorithm, found in [14]. It follows from the matroid properties of minimum spanning trees.

Lemma 1 *Let T be a minimum spanning tree on the terminal set X , and let T' be a spanning tree for X . We can transform T' to T by a series of edge swaps, where each swap involves replacing an edge $e_i \in E(T')$ by $e_j \in E(T)$ such that $\|e_i\| \geq \|e_j\|$.*

The corollary below shows that T is equivalent to any minimum spanning tree on $X \cup S$.

Corollary 2 *If T' is a generalised k -Steiner minimum tree on X with Steiner points S then every minimum spanning tree on $X \cup S$ is a generalised k -Steiner minimum tree on X .*

Proof. Let T be a minimum spanning tree on $X \cup S$. By Lemma 1, we can transform T' to T by a series of edge swaps, each of which replaces an edge with another of the same length. By the symmetry of α each such edge swap does not increase $\|T'\|_\alpha$. ■

Throughout this paper we perform various constructions involving the unit ball B for the given norm $\|\cdot\|$, for instance calculating the intersections of two unit balls. Our main interest in this paper is in the computational nature, specifically the time-complexity, of a solution to any instance of the generalised k -Steiner tree problem. In order to find efficient algorithms, we need to compute these unit ball operations to within any fixed precision in constant time. We therefore restrict the norm $\|\cdot\|$ so that its unit ball is always simple enough to perform these operations. We will provide more detail regarding these restrictions in the next section. For similar computational reasons we will also be placing a restriction on α , and this is discussed in Section 4.

3 The Overlaid Oriented Dirichlet Cell Partition

Let a norm $\|\cdot\|$ on \mathbb{R}^2 be given with corresponding unit ball B . Our aim in this section is to describe the construction of the oriented Dirichlet cell (ODC) partition for any set X of n terminals embedded in this normed plane, and to show that it can be constructed within a time of $\mathcal{O}(n \log n)$. We also show that, with a time complexity of $\mathcal{O}(n^2)$, multiple ODC partitions can be overlaid. This final partition is the afore-mentioned *overlaid ODC partition* (OODC partition), and is a core component of our algorithm.

Georgakopoulos and Papadimitriou [6] allude to a simple method of constructing an ODC partition for terminals embedded in the Euclidean plane. Unfortunately this method does not work for arbitrary normed planes. We circumvent this problem by defining a type of abstract Voronoi diagram that is equivalent to the ODC partition, and then showing that this Voronoi diagram can be calculated in the required time.

We now state the first of three restrictions on B . We defer a discussion of these restrictions (including the description of a class of norms that satisfy all of them) to the end of the section. Let the boundary of B be denoted by $\text{bd}(B)$.

Restriction 1 *The intersection points of any two translated copies of $\text{bd}(B)$, and the intersection points of any straight line and $\text{bd}(B)$, can be calculated to within any fixed precision in constant time.*

Lemma 3 *There exist six points $\{\mathbf{y}_i : i = 0, \dots, 5\}$ on $\text{bd}(B)$ such that for any pair of rotationally consecutive ones, say $\mathbf{y}_i, \mathbf{y}_j$, we have $\|\mathbf{y}_i - \mathbf{y}_j\| = 1$. Moreover, these six points are constructible.*

Proof. The standard ruler and compass construction of the hexagon will produce these six points, where B plays the role of the circle in the construction. Given any point \mathbf{y}_5 on $\text{bd}(B)$ construct a translation of $\text{bd}(B)$ centered around \mathbf{y}_5 . Let \mathbf{y}_0 be the first intersection point of the two boundaries as we traverse the boundary of the original ball anticlockwise from \mathbf{y}_5 . Let $\mathbf{y}_1 = \mathbf{y}_0 - \mathbf{y}_5$. Note that this point also lies on $\text{bd}(B)$ and that $\mathbf{y}_0\mathbf{y}_1\mathbf{o}\mathbf{y}_5$ is a parallelogram. The remaining three points are constructed using the central symmetry of B . See Figure 1 for an example where B is a tilted ellipse. The distance properties of the lemma follow easily by construction. ■

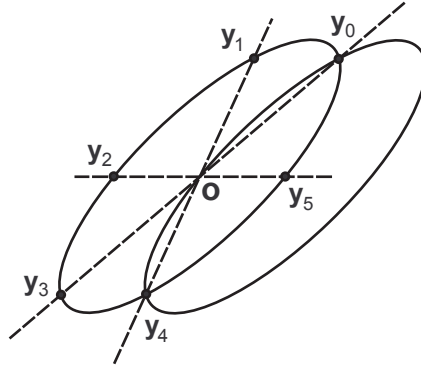


Figure 1: The standard hexagon construction

For any two directions ϕ_i and ϕ_j in the plane, we denote by $K(\mathbf{y}, \phi_i, \phi_j)$ the cone with vertex \mathbf{y} from the ray with vertex y in direction ϕ_i to the ray with vertex y in direction ϕ_j . For each \mathbf{y}_i from Lemma 3 let θ_i be the direction of the ray $\overrightarrow{\mathbf{o}\mathbf{y}_i}$, where \mathbf{o} is the center of B . We assume that the $\{\theta_i\}$ are ordered in an anti-clockwise manner, and two consecutive directions will be denoted by θ_i and θ_{i+1} (i.e. the mod 6 notation will be omitted). As another example we show, in Figure 2, the six directions produced when B is the unit ball of the rectilinear plane.

Lemma 4 *Let $\mathbf{x} \in B$, $\mathbf{x} \neq \mathbf{o}$ and \mathbf{a} and \mathbf{b} points on the boundary of B such that the segments $\mathbf{a}\mathbf{o}$ and $\mathbf{b}\mathbf{x}$ intersect in P . Then $\|\mathbf{a} - \mathbf{x}\| \leq \|\mathbf{b} - \mathbf{x}\|$.*

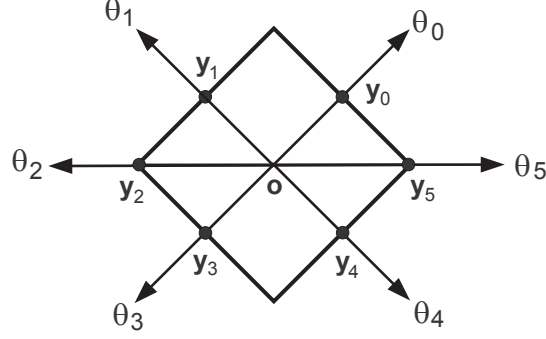


Figure 2: The unit ball of the rectilinear plane and corresponding $\{\theta_i\}$

Proof. Applying the Triangle Inequality to $\triangle \mathbf{pax}$ and $\triangle \mathbf{pbo}$, we obtain $\|\mathbf{a} - \mathbf{x}\| + \|\mathbf{b}\| \leq \|\mathbf{b} - \mathbf{x}\| + \|\mathbf{a}\|$ from which the lemma follows; see Figure 3. ■

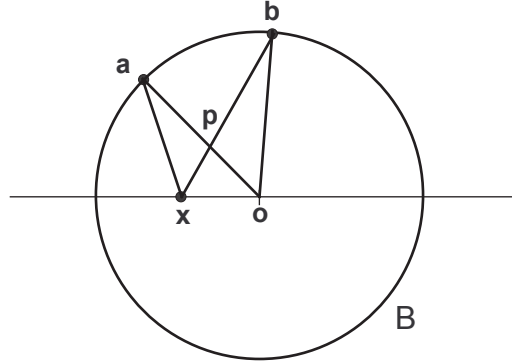


Figure 3: Illustration of the proof of Lemma 4

Lemma 5 *Let \mathbf{y} be any point in the plane. Then there exists a minimum spanning tree T on $X \cup \{\mathbf{y}\}$ with the following property: for each $i = 0, \dots, 5$ there is at most one point of X adjacent to \mathbf{y} in T and lying within $K(\mathbf{y}, \theta_i, \theta_{i+1})$, and this point is a closest terminal to \mathbf{y} in the cone.*

Proof. Let T' be a minimum spanning tree on $X \cup \{\mathbf{y}\}$. Let $\mathbf{x}_0 \in X$ be a terminal in $K(\mathbf{y}, \theta_i, \theta_{i+1})$ that is closest to \mathbf{y} , and suppose that $(\mathbf{y}, \mathbf{x}_1) \in E(T')$ where $\mathbf{x}_1 \in X$ is any other terminal in $K(\mathbf{y}, \theta_i, \theta_{i+1})$. We show that we can replace the edge $(\mathbf{y}, \mathbf{x}_1)$ in T' by either $(\mathbf{y}, \mathbf{x}_0)$ or $(\mathbf{x}_1, \mathbf{x}_0)$ so that the resulting tree is still a minimum spanning tree on $X \cup \{\mathbf{y}\}$. From this, the statement of the lemma follows.

Assume first that the path in T' connecting \mathbf{y} and \mathbf{x}_0 passes through \mathbf{x}_1 . In this case we can replace $(\mathbf{y}, \mathbf{x}_1)$ by $(\mathbf{y}, \mathbf{x}_0)$ without losing connectivity or increasing the length of T' .

Assume, on the other hand, that the path in T' connecting \mathbf{y} and \mathbf{x}_0 does not pass through \mathbf{x}_1 .

Claim: $\|\mathbf{x}_0 - \mathbf{x}_1\| \leq \|\mathbf{y} - \mathbf{x}_1\|$.

Without loss of generality, we assume that $\mathbf{y} = \mathbf{o}$, $\|\mathbf{x}_1\| = 1$, and $K(\mathbf{o}, \theta_i, \theta_{i+1})$ intersects the boundary of the unit ball B in an arc from \mathbf{a} to \mathbf{b} (with $\|\mathbf{a} - \mathbf{b}\| = 1$). We can also assume, without loss of generality, that \mathbf{x}_1 lies on the same side of the line through $\mathbf{o}\mathbf{x}_0$ as \mathbf{b} . The convexity of B implies that the line segments $\mathbf{o}\mathbf{x}_1$ and $\mathbf{a}\mathbf{b}$ intersect, hence by Lemma 4 we have

$$\|\mathbf{a} - \mathbf{x}_1\| \leq \|\mathbf{a} - \mathbf{b}\| = 1. \quad (1)$$

We now prove the claim via two cases, illustrated in Figure 4. Firstly, suppose \mathbf{x}_0 and \mathbf{o}

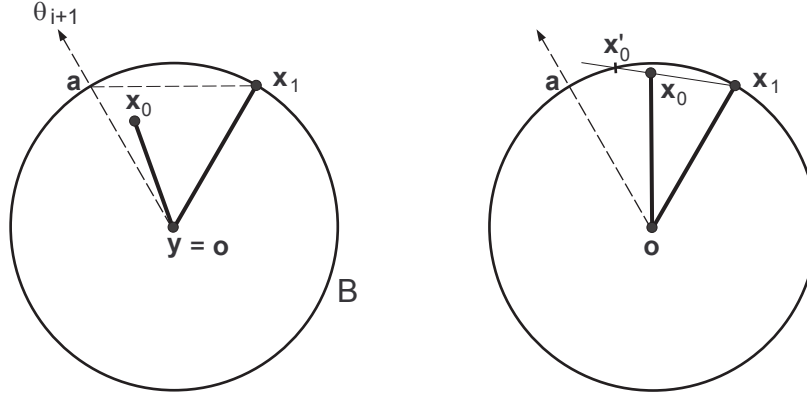


Figure 4: The two cases of the Claim in the proof of Lemma 5

are on the same side of $\mathbf{a}\mathbf{x}_1$ (including the case where \mathbf{x}_0 lies on $\mathbf{a}\mathbf{x}_1$). By Inequality (1) \mathbf{a} lies in the unit ball centered at \mathbf{x}_1 , so, by convexity, \mathbf{x}_0 also lies in this unit ball. Hence, $\|\mathbf{x}_0 - \mathbf{x}_1\| \leq 1$ as required. For the second case, suppose that \mathbf{x}_0 and \mathbf{o} are on opposite sides of $\mathbf{a}\mathbf{x}_1$. Let the ray from \mathbf{x}_1 passing through \mathbf{x}_0 intersect B at \mathbf{x}'_0 . Then \mathbf{x}'_0 and \mathbf{o} are also on opposite sides of $\mathbf{a}\mathbf{x}_1$, and hence, by Lemma 4, $\|\mathbf{x}_0 - \mathbf{x}_1\| \leq \|\mathbf{x}'_0 - \mathbf{x}_1\| \leq \|\mathbf{a} - \mathbf{x}_1\|$. Therefore, $\|\mathbf{x}_0 - \mathbf{x}_1\| \leq 1$ by Inequality (1), and the claim is proven.

By the claim we can now replace the edge $(\mathbf{y}, \mathbf{x}_1)$ by $(\mathbf{x}_1, \mathbf{x}_0)$ without losing connectivity or increasing the length of T' . ■

For each $i = 0, \dots, 5$, the i th *oriented Dirichlet cell* (ODC) of $\mathbf{w} \in X$ is the set:

$$\{\mathbf{y} \in \mathbb{R}^2 : \|\mathbf{w} - \mathbf{y}\| = \min\{\|\mathbf{x} - \mathbf{y}\|, \mathbf{x} \in X \cap K(\mathbf{y}, \theta_i, \theta_{i+1})\}\}$$

In other words, this is the set of all points $\{\mathbf{y}\}$ whose closest terminal in the cone $K(\mathbf{y}, \theta_i, \theta_{i+1})$ is \mathbf{w} . We will show that the set of i th ODCs, called the i th *ODC partition of X* is a type of Voronoi diagram.

In [2] Chew and Drysdale present an “expanding waves” view of Voronoi diagrams. If n pebbles are dropped simultaneously into a pond, the places where wave fronts meet define

the Voronoi diagram on the n points of impact. In the Euclidean case the wavefronts are circular, but in theory any closed convex oriented curve can qualify as a wavefront and thereby define an abstract Voronoi diagram. For any such shape C and set of terminals X we say that the resulting diagram is the *Voronoi diagram of X based on C* .

We may define this Voronoi diagram more formally as follows. Let $\delta_C : \mathbb{R}^2 \rightarrow \mathbb{R}$ be the distance function based on C ; in other words, for any points $\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^2$ we let $\delta_C(\mathbf{x}_0, \mathbf{x}_1) = \inf\{\lambda^{-1} : \lambda(\mathbf{x}_1 - \mathbf{x}_0) \in C\}$. We then define a region $V_{\mathbf{x}} = \{\mathbf{y} : \delta_C(\mathbf{x}, \mathbf{y}) \in \mathbb{R} \text{ and } \delta_C(\mathbf{x}, \mathbf{y}) = \min\{\delta_C(\mathbf{x}', \mathbf{y}) : \mathbf{x}' \in X\}\}$ for each $\mathbf{x} \in X$. The set $\{V_{\mathbf{x}}\}$ is the required Voronoi diagram based on C .

Proposition 6 *For any $i = 0, \dots, 5$ the i th ODC partition of X is equal to the Voronoi diagram of X based on the sector $B \cap K(\mathbf{o}, 180^\circ + \theta_i, 180^\circ + \theta_{i+1})$.*

Proof. This follows immediately from the central symmetry of B . ■

In Figure 5 we give an example of an ODC partition when the original unit ball is a circle (i.e. the Euclidean case) and therefore C is a circular sector, and in Figure 6 we give an example of what the boundary of wavefronts look like when C is a regular hexagon.

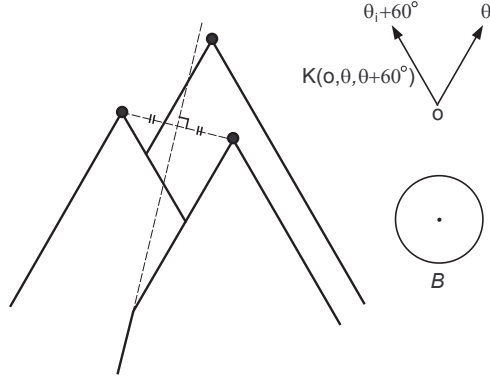


Figure 5: An ODC partition of a three-terminal example

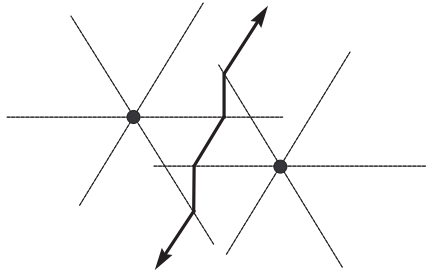


Figure 6: Regular hexagon based Voronoi diagram for two points

The next theorem now gives us the required time-complexity for calculating the i th ODC partition under certain conditions.

Theorem 7 [2] *The Voronoi diagram of n points based on a closed convex shape C can be constructed in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space as long as the following operations can be performed in constant time:*

1. *Given two points, find the boundary where the two wavefronts meet.*
2. *Given two such boundaries, compute their intersection(s).*

We therefore also impose the following restriction on B .

Restriction 2 *Let C' be any sector of B . Then, given any two points, we can find the boundary where the two wavefronts based on C' meet, and, given two such boundaries, we can compute their intersection. Moreover, these operations can be performed to within any fixed precision in constant time.*

The next step is to overlay the six i th ODC partitions. The theorem we use, which is a result from [6], assumes that the regions in each partition have boundaries consisting of straight line segments. We therefore state our third and final restriction on B .

Restriction 3 *The shape of B implies that the i th ODC partition of any set of points is linear (i.e. the boundary of any ODC consists of straight line segments).*

Theorem 8 [6] *Let q be any positive integer. Then q linear plane partitions can be overlaid in $\mathcal{O}(q^2 n^2)$ time, where n is the total number of regions in each partition.*

As a consequence of the previous results, within $\mathcal{O}(n \log n)$ total time we can calculate the i th ODC partition of the plane for each i , and then, in a time of $\mathcal{O}(n^2)$, overlay these six partitions resulting in the OODC partition. It is easily observed that the OODC partition has $\mathcal{O}(n^2)$ regions.

Let R be a region of the OODC partition. Note that R is the intersection of at most six ODCs. Let $\{D_j\}$ be the set of these ODCs, where D_j is an ODC of $p_j \in X$. Let $C_X(R) = \{p_j\}$. The power of the OODC partition lies in the next theorem, which now follows from Lemma 5.

Theorem 9 *Let \mathbf{s} be any point in R . Then there exists a minimum spanning tree T on $X \cup \{\mathbf{s}\}$ such that the set of neighbours of \mathbf{s} in T is a subset of $C_X(R)$.*

The question arises as to whether norms exist with unit balls satisfying all three restrictions. Let \mathcal{V} be the class of norms defined by the condition that each norm's unit ball is either a polygon or an ellipse. Suppose that $\|\cdot\| \in \mathcal{V}$ and that the corresponding unit ball is B . Clearly Restriction 1 is true for B . Given any two points, their bisector (based

on a sector of B) will be a polygonal line that can be computed with some simple vector operations (see [2]). The same holds true for intersecting two boundaries, and therefore Restriction 2 is satisfied. Since any i th ODC partition consists of segments of bisectors and segments of the limiting rays of $K(\mathbf{y}, \theta_i, \theta_{i+1})$ for some \mathbf{y} , Restriction 3 follows immediately.

4 Generalised Steiner Tree Problems for a Fixed Topology

Each main iteration of our algorithm produces a set of neighbours (from $X \cup S$) for each of the $k' \leq k$ Steiner points. Finding the optimal coordinates of the Steiner points for the subtree induced by the Steiner points and their neighbours is a problem known in the literature as the *fixed topology Steiner tree problem*. We state the problem more formally as follows: given a set X' of $c \leq 6k'$ embedded terminals, a set S of k' free (i.e. non-embedded) Steiner points, and a tree topology \mathcal{T} spanning all these nodes, we wish to find the coordinates of the Steiner points (i.e. find the set S) such that $\alpha(\mathbf{e}_{\mathcal{T}, X', S})$ is minimised, where $\mathbf{e}_{\mathcal{T}, X', S} = (\|e_1\|, \dots, \|e_{c+k'-1}\|)$. For our purposes a node x of \mathcal{T} will be of degree one if and only if $x \in X'$. This problem is interesting in its own right, but is also a key step of our main algorithm. We therefore restrict α so that this problem is always computable to within any fixed precision in constant time, for a fixed value of k . In fact, k (and therefore c) will generally be much smaller than n , and therefore we are not particularly interested in the time-complexity of this step.

We discuss briefly a few functions that satisfy this restriction.

- (1) $\alpha(\mathbf{e}_{\mathcal{T}, X', S}) = \sum \|e_i\|$. In this case we are dealing with the well-known Steiner tree problem for a fixed topology. In the Euclidean plane the problem has an $\mathcal{O}(c^2)$ -time solution provided that no point has degree larger than 3, see [10]. Unfortunately, for the k -Steiner tree problem degree 4 points do exist (but degree 5 do not; see [18]), and this more general problem may be NP-complete. A similar result holds for the rectilinear and other fixed orientation planes [1].
- (2) $\alpha(\mathbf{e}_{\mathcal{T}, X', S}) = \sum \|e_i\|^p$, $p > 0$. This is referred to as the power- p Steiner tree problem for a fixed topology. In the Euclidean plane with $p = 2$, Ganley [4] shows that the problem can be solved within time $\mathcal{O}(c)$.
- (3) $\alpha(\mathbf{e}_{\mathcal{T}, X', S}) = \lim_{p \rightarrow \infty} \left(\sum \|e_i\|^p \right)^{1/p}$, i.e. the bottleneck Steiner problem for a fixed topology. This problem has an $\mathcal{O}(c^2)$ solution in the rectilinear plane, see [5]. In the Euclidean and general ℓ_p planes there exists various numerical algorithms that can calculate a solution to any desired precision, see for instance [3], [16]. A fully polynomial time approximation scheme (FPTAS) exists for the problem in the Euclidean plane (see [4]).

5 Minimum F -fixed Spanning Trees

Many papers exist in the literature that deal with the time-complexity of updating a minimum spanning tree when a new node is introduced; see for instance [11] where they show that a tree on n nodes can be updated with a new node in $\mathcal{O}(\log n)$ parallel time using $n/\log n$ exclusive read, exclusive write, parallel random access machines (EREW PRAMs). Georgakopoulos and Papadimitriou utilise a preprocessing step in [6] so that a minimum spanning tree can be updated in constant time with a new point.

In order for our algorithm to find an optimal tree for all Steiner points simultaneously, it must update a minimum spanning tree (in constant time) to include a new subforest F . However, it is not necessary for the updated tree T_F to be a minimum spanning tree on its nodes, but only that it is a shortest total length tree spanning $X \cup V(F)$ such that the neighbour-set of each Steiner point in F remains unchanged in T_F .

Let $N(T, s)$ denote the set of neighbours of a node s in a graph T . A forest F with node-set $X' \cup S$, where $X' \subseteq X$ and $S \subseteq \mathbb{R}^2$ with $|S| \leq k$, is called *feasible* if and only if $\{\mathbf{x} \in V(F) : \mathbf{x} \text{ is a leaf of } F\} = X'$ and $|N(F, \mathbf{s})| \leq 6$ for every $\mathbf{s} \in S$. If S is the set of at most k new embedded Steiner points then, in our main algorithm, the forest F induced by the set of new edges incident to elements of S will be feasible. A shortest total length tree T_F , such that $V(T_F) = X \cup S$ and $N(T_F, \mathbf{s}) = N(F, \mathbf{s})$ for every $\mathbf{s} \in S$, is referred to as a *minimum F -fixed spanning tree*.

We use the symbol $P_T(\mathbf{x}, \mathbf{y})$ to represent a path through T with endpoints \mathbf{x} and \mathbf{y} , and we use $\ell_T(\mathbf{x}, \mathbf{y})$ to denote the longest edge on $P_T(\mathbf{x}, \mathbf{y})$. We will make use of the following theorem.

Theorem 10 [13] *A tree T is a minimum spanning tree on X if and only if for every pair of non-adjacent nodes $\mathbf{x}, \mathbf{y} \in X$, $\|e\| \leq \|\mathbf{x} - \mathbf{y}\|$ for every $e \in E(P_T(\mathbf{x}, \mathbf{y}))$.*

Let T now be a minimum spanning tree on X . The preprocessing stage PS1 calculates $\ell_T(\mathbf{x}, \mathbf{y})$ for every pair of nodes $\mathbf{x}, \mathbf{y} \in V(T)$. This requires $\mathcal{O}(n^2)$ time and $\mathcal{O}(n^2)$ space. We incorporate a consistent tie-breaking component for choosing between edges of exactly the same length by placing an order on $E(T)$ and choosing the earlier edge in this ordering whenever a tie occurs.

Proposition 11 *Let T be a minimum spanning tree on X , and assume that the above preprocessing step has been performed. If F is connected and feasible, then a minimum F -fixed spanning tree T_F can be constructed from T in $\mathcal{O}(k^2)$ time.*

Proof. Let $G = T \cup F$, let $A = V(F) \cap X$, and note that $|A| \leq 6k$. A number of cycles may occur in G , each one of them containing a path through F with endpoints from A . Let T' be the graph obtained by deleting the set of edges $\{\ell_T(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i, \mathbf{x}_j \in A, i \neq j\}$ from G . We will show that $T' = T_F$, which suffices to prove the proposition since T' can clearly be constructed in $\mathcal{O}(k^2)$ time.

To prove that $T' = T_F$ we first show that T' is acyclic and spans $X \cup S$. Every cycle of G is of the form $P_F(\mathbf{x}_i, \mathbf{x}_j), P_T(\mathbf{x}_j, \mathbf{x}_i)$, and therefore deleting every $\ell_T(\mathbf{x}_i, \mathbf{x}_j)$ from G produces an acyclic graph. We use induction on $|A|$ to prove that T' is connected. Let $A_b = \{\mathbf{x}_1, \dots, \mathbf{x}_b\} \subseteq A$ for some $b \in \{2, \dots, 6k\}$, let F_b be the subtree of F induced by $S \cup A_b$, and let $G_b = T \cup F_b$. Subtracting $L_b = \{\ell_T(\mathbf{x}_i, \mathbf{x}_j) : 1 \leq i < j \leq b\}$ from $E(G_b)$ produces the graph T_b . For the base case we let $b = 2$. The only cycle of G_2 is $P_F(\mathbf{x}_1, \mathbf{x}_2), P_T(\mathbf{x}_2, \mathbf{x}_1)$, and $\ell_T(\mathbf{x}_2, \mathbf{x}_1)$ is an edge of this cycle. Therefore deleting $\ell_T(\mathbf{x}_2, \mathbf{x}_1)$ does not destroy the connectivity of T_2 on $X \cup S$.

Next assume that T_b spans $X \cup S$ for some $2 \leq b \leq 6k - 1$ and suppose that $\mathbf{x}_{b+1} \in X \setminus A_b$. Since T_b is connected and acyclic there is exactly one path connecting \mathbf{x}_{b+1} to a node of A_b not passing through any element of S , i.e. this path is of the form $P_T(\mathbf{x}_{b+1}, \mathbf{x}_r)$ for some unique $\mathbf{x}_r \in A_b$. Let $A_{b+1} = A_b \cup \{\mathbf{x}_{b+1}\}$ and let $\mathbf{s} = N(F, \mathbf{x}_{b+1})$. Then T_{b+1} is the graph with $V(T_{b+1}) = X \cup S$ and $E(T_{b+1}) = (E(T_b) \cup \{(\mathbf{s}, \mathbf{x}_{b+1})\}) \setminus \{\ell_T(\mathbf{x}_{b+1}, \mathbf{x}_i) : \mathbf{x}_i \in A_b\}$.

Claim: For every $\mathbf{x}_i \in A_b$ either $\ell_T(\mathbf{x}_{b+1}, \mathbf{x}_i) \in L_b$ or $\ell_T(\mathbf{x}_{b+1}, \mathbf{x}_i) = \ell_T(\mathbf{x}_{b+1}, \mathbf{x}_r)$. Let $\mathbf{x}_i \in A_b \setminus \{\mathbf{x}_r\}$ and consider the following two cases. If \mathbf{x}_{b+1} lies on $P_T(\mathbf{x}_i, \mathbf{x}_r)$ then $\ell_T(\mathbf{x}_i, \mathbf{x}_r) = \max\{\ell_T(\mathbf{x}_i, \mathbf{x}_{b+1}), \ell_T(\mathbf{x}_{b+1}, \mathbf{x}_r)\} = \ell_T(\mathbf{x}_i, \mathbf{x}_{b+1})$ since $P_T(\mathbf{x}_{b+1}, \mathbf{x}_r)$ is a path in T_b and therefore does not contain $\ell_T(\mathbf{x}_i, \mathbf{x}_r)$. Therefore $\ell_T(\mathbf{x}_{b+1}, \mathbf{x}_i) \in L_b$. For the second case, if \mathbf{x}_{b+1} does not lie on $P_T(\mathbf{x}_i, \mathbf{x}_r)$ then let \mathbf{y} be the first common point of the paths $P_T(\mathbf{x}_i, \mathbf{x}_r)$ and $P_T(\mathbf{x}_{b+1}, \mathbf{x}_r)$; see Figure 7.

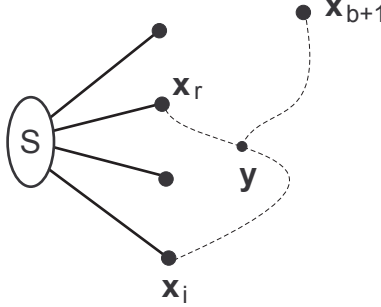


Figure 7: The second case of the claim

Note that \mathbf{y} may be equal to \mathbf{x}_r . Clearly

$$\ell_T(\mathbf{x}_i, \mathbf{y}) \geq \ell_T(\mathbf{y}, \mathbf{x}_r) \quad (2)$$

since $P_T(\mathbf{y}, \mathbf{x}_r)$ is also a path in T_b . There are now two possibilities to consider; either $\ell_T(\mathbf{x}_{b+1}, \mathbf{x}_i) = \ell_T(\mathbf{x}_i, \mathbf{y}) = \ell_T(\mathbf{x}_i, \mathbf{x}_r) \in L_b$, or $\ell_T(\mathbf{x}_{b+1}, \mathbf{x}_i) = \ell_T(\mathbf{x}_{b+1}, \mathbf{y}) = \ell_T(\mathbf{x}_{b+1}, \mathbf{x}_r)$, where for each possibility the second equality follows from Inequality (2). The claim follows.

By the above claim $E(T_{b+1}) = (E(T_b) \cup \{(\mathbf{s}, \mathbf{x}_{b+1})\}) \setminus \{\ell_T(\mathbf{x}_{b+1}, \mathbf{x}_r)\}$ and we deduce that T_{b+1} has been constructed from T_b by adding one edge from F and then deleting an edge of T on the resultant cycle. This completes the induction argument, and hence T' is connected and spans $X \cup S$.

Next we prove that T' is a *minimum* F -fixed spanning tree. Let K be the complete graph on X . Furthermore, suppose that the edges of K are weighted by the function w , where

$$w((\mathbf{x}, \mathbf{y})) = \begin{cases} 0 & \text{if } \mathbf{x} \in A \text{ and } \mathbf{y} \in A, \\ \|\mathbf{x} - \mathbf{y}\| & \text{otherwise.} \end{cases}$$

Let T_A be any spanning tree on A . Then clearly T' is a minimum F -fixed spanning tree if and only if the graph T_K , where $V(T_K) = X$ and $E(T_K) = (E(T') \cap (X \times X)) \cup T_A$, is a minimum spanning tree of K with the above weight function. But this follows from a simple application of Theorem 10. Hence $T' = T_F$, as required. ■

An immediate consequence of the above proof is the following result.

Corollary 12 $|\{\ell_T(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i, \mathbf{x}_j \in A\}| = |A| - 1$.

Next we generalise the Proposition 11 to the case where F is not necessarily connected, which must be considered if $k > 1$. If $k > 1$ we perform an additional preprocessing stage, PS2, to calculate a TRUE/FALSE table H such that $H_{e, \mathbf{y}, \mathbf{z}} = \text{TRUE}$ if and only if edge $e \in E(P_T(\mathbf{y}, \mathbf{z}))$. This requires at most $\mathcal{O}(n^2)$ time and $\mathcal{O}(n^3)$ space. For each connected component F^i of F let $A^i = V(F^i) \cap X$. We claim that *Algorithm F-MST*, given in Table 1, calculates a minimum F -fixed spanning tree for any feasible forest F .

In essence, Algorithm F -MST deletes the longest edge not in F , of every cycle in $T \cup F$. The shortest path through J connecting some pair $\mathbf{x}, \mathbf{y} \in A^i$ is simply a representation of the sequence of paths through T that are contained in $P_{T^{i-1}}(\mathbf{x}, \mathbf{y})$. The longest edge of all paths in this sequence is therefore the longest edge on the path $P_{T^{i-1}}(\mathbf{x}, \mathbf{y})$ barring any edges of F . It should be clear that any inductive method that correctly deletes the longest edge of each newly created cycle, is equivalent to deleting all longest cycle-edges simultaneously. To formally prove correctness of Algorithm F -MST we need the following lemma.

Lemma 13 *Let $\mathbf{x}, \mathbf{y} \in A^i$. Then $\ell^i(\mathbf{x}, \mathbf{y})$ is the longest edge in $E(P_{T^{i-1}}(\mathbf{x}, \mathbf{y})) \setminus \bigcup_{j=1}^{i-1} E(F^j)$.*

Proof. Let J be the graph defined in Step 1a(i) of Algorithm F -MST, and let $P' = \{\mathbf{x}\}, A^{b_1}, \dots, A^{b_d}, \{\mathbf{y}\}$ be any path in J , where clearly $1 \leq b_j \leq i-1$. Note first that σ_1 and σ_2 are well defined functions on $E(J)$ since, as we will show in the next proposition, T^{i-1} is a tree. If e_1, \dots, e_{d+1} is the edge-sequence of P' then there is a corresponding $\mathbf{x} - \mathbf{y}$ walk through T^{i-1} of the form

$$W(P') = P_T(\sigma_1(e_1), \sigma_2(e_1)), P^{b_1}, P_T(\sigma_1(e_2), \sigma_2(e_2)), P^{b_2}, \dots, P_T(\sigma_1(e_{d+1}), \sigma_2(e_{d+1}))$$

where: $\sigma_1(e_1) = \mathbf{x}$; $\sigma_2(e_{d+1}) = \mathbf{y}$; $\sigma_1(e_j) \in A^{b_{j-1}}$ for $1 < j \leq d+1$; $\sigma_2(e_j) \in A^{b_j}$ for $1 \leq j \leq d$; every P^{b_j} is a path in F^{b_j} between two of its leaves, or is the empty set.

Algorithm <i>F</i> -MST	
Input: A set X of points in the plane, a minimum spanning tree T on X , and a feasible forest F with t connected components. Output: A minimum F -fixed spanning tree T_F on $X \cup V(F)$.	
Step	Description
	Let $D^0 = \emptyset$ and let $T^0 = T$.
1	For $i = 1$ to t Do
	Begin
1a	For every distinct pair $\mathbf{x}, \mathbf{y} \in A^i$ Do
	Begin
1a(i)	Let J be the graph with $V(J) = \begin{cases} \{\{\mathbf{x}\}, \{\mathbf{y}\}\} & \text{if } i = 1, \\ \{\{\mathbf{x}\}, \{\mathbf{y}\}, A^1, \dots, A^{i-1}\} & \text{if } i > 1. \end{cases}$ and $E(J) = \{(U, U') : \exists \mathbf{w} \in U \wedge \exists \mathbf{w}' \in U' \text{ such that } H_{e, \mathbf{w}, \mathbf{w}'} = \text{FALSE } \forall e \in D^{i-1}\}.$
1a(ii)	For every $e = (U, U') \in E(J)$ let $\sigma_1(e) = \mathbf{w}$ and let $\sigma_2(e) = \mathbf{w}'$ (where \mathbf{w}, \mathbf{w}' are from the previous step).
1a(iii)	Perform a search through J to find the path $P = P_J(\{\mathbf{x}\}, \{\mathbf{y}\})$ such that $ E(P) $ is a minimum. Let $\ell^i(\mathbf{x}, \mathbf{y})$ be the edge from $\{\ell_T(\sigma_1(e), \sigma_2(e)) : e \in E(P)\}$ of maximum length.
	End
	Let $L^i = \{\ell^i(\mathbf{x}, \mathbf{y}) : \mathbf{x}, \mathbf{y} \in A^i\},$ $D^i = D^{i-1} \cup L^i,$ $G^i = T^{i-1} \cup F^i,$ $T^i = \langle V(G^i), E(G^i) \setminus D^i \rangle.$
	End
	Let $T_F = T^t$.

Table 1: Algorithm *F*-MST

Let W' be any $\mathbf{x} - \mathbf{y}$ walk through T^{i-1} , and suppose that as we trace W' from \mathbf{x} to \mathbf{y} , we consecutively intersect Steiner points of the distinct trees F^{a_1}, \dots, F^{a_r} , where each $1 \leq a_j \leq i-1$. Let $W^{-1}(W') = \{\mathbf{x}\}, A^{a_1}, \dots, A^{a_r}, \{\mathbf{y}\}$. Note that it is not necessarily the case that $W^{-1}(W(P')) = P'$. Now suppose that $P = W^{-1}(P_{T^{i-1}}(\mathbf{x}, \mathbf{y})) = \{\mathbf{x}\}, A^{a_1}, \dots, A^{a_r}, \{\mathbf{y}\}$ (which, note, is a path through J). Since $W(P')$ contains the unique path $P_{T^{i-1}}(\mathbf{x}, \mathbf{y})$, the node-sequence of P must be a subsequence of the node-sequence of P' . Since this holds for any P' , this implies that P is the path through J of minimum length (i.e. minimising $|E(P)|$) connecting $\{\mathbf{x}\}$ and $\{\mathbf{y}\}$. The lemma follows. ■

Proposition 14 *Let T be a minimum spanning tree on X , and assume that preprocessing steps PS1 and PS2 have been performed. If F is a feasible forest then Algorithm F -MST correctly produces a minimum F -fixed spanning tree T_F in $\mathcal{O}(k^{2k+3}k!)$ time.*

Proof. The proposition is easily verified by using induction on t (the number of connected components of F). Proposition 11 proves the base case $t = 1$, and similar reasoning is used to prove that each subsequent T^i is connected. At each inductive step, Lemma 13 assures us that Algorithm F -MST correctly deletes the longest edge (excluding edges of F) of any new cycle formed. To prove minimality of T_F we once again construct (analogously to Proposition 11) a weighted complete graph K on X and a tree T_K , such that T_F is a minimum F -fixed spanning tree on $X \cup V(F)$ if and only if T_K is a minimum spanning tree of K . Theorem 10 then completes the minimality proof.

To verify the time-complexity first note that $t \leq k$ and $|A^i| \leq 6k$. Step 1a of the algorithm requires $\mathcal{O}(k^2)$ time, Step 1a(i) requires $\mathcal{O}(k^{2k}k!)$ time and Step 1a(iii) requires $\mathcal{O}(k)$ time. ■

Recall that when $k = 1$ we do not need to calculate the table H during preprocessing because Proposition 11 will apply. Therefore the space complexity reduces to $\mathcal{O}(n^2)$ in this case.

6 The Main Algorithm

We present *Algorithm k -GSMT*, in Table 2.

By using Cayley's formula and the observation that each spanning forest is a subgraph of a spanning tree which has $k - 1$ edges, we get an upper bound for $f_1(k)$ of $126^k \cdot k^{k-2}$ in Step 4b of the algorithm. The function f_2 in Step 4b(i) will depend on the relevant generalised k -Steiner tree problem. We mention once again that we consider k to be much smaller than n , and therefore we conclude that the overall time-complexity of Algorithm k -GSMT is $\mathcal{O}(n^{2k})$.

We now demonstrate the correctness of the algorithm. Let T_{opt} be a generalised k -Steiner minimum tree on X . Let S be the set of Steiner points in T_{opt} and let F_{opt} be the sub-forest of T_{opt} induced by the edges of T_{opt} incident with elements of S . Let F_{opt}^i be a connected

Algorithm k -GSMT		
Input: A set X of n points in the plane, a unit ball B , a positive integer k , and a symmetric ℓ_1 -optimisable function α .		
Output: A set S of at most k Steiner points, and a tree T^* interconnecting $X \cup S$, such that $\ T^*\ _\alpha = \min_{\mathcal{T}, S'} \alpha(\mathbf{e}_{\mathcal{T}, X, S'})$.		
Step	Description	Time
1	Construct the OODC partition of X .	$\mathcal{O}(n \log n)$
2	Construct a minimum spanning tree T on X .	$\mathcal{O}(n \log n)$
3	Perform preprocessing steps PS1 and PS2 on T .	$\mathcal{O}(n^2)$
4	For every $k' \leq k$ and each choice (with repetition) of k' regions, $R_1, \dots, R_{k'}$, of the OODC partition Do	$\mathcal{O}(n^{2k})$
	Begin	
4a	Associate the free Steiner point s_i with region R_i . Let \mathcal{G} be the graph consisting of the vertices $\bigcup C_X(R_i) \cup \{s_1, \dots, s_{k'}\}$, all edges $(s_i, s_j), i \neq j$, and all edges (s_i, \mathbf{x}) for every $\mathbf{x} \in C_X(R_i)$. Let \mathcal{G}^* be the set of all feasible subforests of \mathcal{G} .	
4b	For each $\mathcal{F} \in \mathcal{G}^*$ Do	$\mathcal{O}(f_1(k))$
	Begin	
4b(i)	Solve the fixed topology generalised Steiner tree problem for \mathcal{F} to get the tree F .	$\mathcal{O}(f_2(k))$
4b(ii)	Run Algorithm F -MST with input T and F , and let T_F be its output.	$\mathcal{O}(k^{2k+3}k!)$
	End	
	End	
5	Select a smallest total cost T_F produced and let $T^* = T_F$. Let S be the set of Steiner points of T^* .	

Table 2: Algorithm k -GSMT

component of F_{opt} with k_i Steiner points and terminal set $A^i \subseteq X$. Note that, like T_{opt} , F_{opt} may not be unique for a given set X . We have the following two easy lemmas.

Lemma 15 F_{opt}^i is a generalised k_i -Steiner minimum tree on A^i .

Lemma 16 Let Y^i be any generalised k_i -Steiner minimum tree on A^i . Suppose we transform T_{opt} to T' by replacing the subtree F_{opt}^i on T by Y^i . Then T' is also a generalised k -Steiner minimum tree on X .

Proposition 17 Algorithm k -GSMT constructs a tree T^* that is a generalised k -Steiner minimum tree on the terminal set X .

Proof. By the properties of the OODC partition, during the course of the algorithm a forest F induced by edges incident with Steiner points is constructed with connected components F^i such that each F^i has the same terminal set (i.e. A^i) and the same topology as F_{opt}^i , and therefore

$$\|F^i\|_{\alpha} = \|F_{\text{opt}}^i\|_{\alpha}. \quad (3)$$

We now consider two cases.

Suppose, for the first case, that $F_{\text{opt}} = F$. Step 4b(ii) of the algorithm constructs a minimum F -fixed spanning tree T_F on $X \cup S$. Since T_{opt} is a minimum spanning tree on $X \cup S$ and contains F as a sub-forest, it follows that T_F is also a minimum spanning tree on $X \cup S$. By Lemma 2, $T^* = T_F$ is a generalised k -Steiner minimum tree on X , as required.

If, on the other hand, $F_{\text{opt}} \neq F$, then there is a tree T_F constructed in Step 4b(ii) of the algorithm that is the same as in the previous paragraph, except each F^i is replaced by F_{opt}^i . By Equation (3) and Lemma 16 it again follows that $T^* = T_F$ is a generalised k -Steiner minimum tree on X . ■

7 Conclusion

The outcome of this paper is a generalisation, on multiple fronts, of Georgakopoulos and Papadimitriou's $\mathcal{O}(n^2)$ solution to the 1-Steiner tree problem. By utilising abstract Voronoi diagrams, we build on their complexity-reducing concept of oriented Dirichlet cell partitions. The result is a broadening of the scope of these partitions to include terminal sets in arbitrary normed planes. A bigger challenge in our research was to construct a generalisation to k Steiner points. We achieve this by producing a novel method of updating a minimum spanning tree to include a fixed subtree. A two-part preprocessing stage allows this to be done in constant time with respect to the total number of terminals. One of the key observations of our research was that the main algorithm from [6] basically pertains to any ‘‘Steiner-like’’ problem with cost function α , as long as it is guaranteed that a solution exists which is optimal with respect to α and is also a minimum spanning tree on its complete set of

nodes. This fact allows us to accommodate the class of generalised k -Steiner tree problems with symmetric ℓ_1 -optimisable cost functions. The result is an $\mathcal{O}(n^{2k})$ -time solution to this class of problems.

It may be possible to generalise our algorithm to higher dimensional spaces, at least in the Euclidean case. A natural starting point could be Monma and Suris's paper [17], where a partition of d -dimensional Euclidean space is constructed that has similar topology-limiting properties as the oriented Dirichlet cell partition. Another future research topic is an analysis of the practicality of implementing our algorithm for the Euclidean 1-bottleneck Steiner tree problem. We are also considering the merits of utilising this algorithm as an iterative heuristic for the Euclidean k -bottleneck Steiner tree problem. Our interest in this topic stems from its application to the deployment of wireless sensor networks.

References

- [1] M. Brazil, M. Zachariasen, Steiner Trees for Fixed Orientation Metrics, *Journal of Global Optimization*, 43 (2009), pp. 141–169.
- [2] L. P. Chew, R. L. Drysdale, III, Voronoi Diagrams Based on Convex Distance Functions, *Proceedings 1st ACM Symposium on Computational Geometry*, (1985), pp. 235–244.
- [3] Z. Drezner, G. O. Wesolowsky, A New method for the Multifacility Minimax Location Problem, *The Journal of the Operational Research Society*, 29 (1978), pp. 1095–1101.
- [4] J. L. Ganley, Geometric Interconnection and Placement Algorithms, Ph. D Thesis, Department of Computer Science, University of Virginia, Charlottesville, VA, 1995.
- [5] J. L. Ganley, J. S. Salowe, Optimal and Approximate Bottleneck Steiner trees, *Operations Research Letters*, 19 (1996), pp. 217–224.
- [6] G. Georgakopoulos, C. H. Papadimitriou, The 1-Steiner Tree Problem, *Journal of Algorithms*, 8 (1987), pp. 122–130.
- [7] E. N. Gilbert, H. O. Pollak, Steiner minimal trees, *SIAM J. Appl. Math.*, 16 (1968), pp. 1–29.
- [8] J. Griffith, G. Robins, J. S. Salowe, T. Zhang, Closing the Gap: Near-Optimal Steiner Trees in Polynomial Time, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13, (1994), pp. 1351–1365.
- [9] F. K. Hwang, D. S. Richards, P. Winter, The Steiner Tree Problem. *Annals of Discrete Mathematics* 53, Elsevier Science Publishers B.V., Amsterdam, 1992.
- [10] F. K. Hwang, J. F. Weng, The Shortest Network under a Given Topology, *Journal of Algorithms*, 13 (1992), pp. 468–488.
- [11] D. B. Johnson, P. Metaxas, Optimal algorithms for the vertex updating problem of a minimum spanning tree, *Proceedings of the Sixth International Parallel Processing Symposium*, March 1992, pp. 306–314.

- [12] A. B. Kahng, G. Robins, A New Class of Iterative Steiner tree Heuristics with Good Performance, *IEEE Transactions on Computer-aided Design*, 11 (1992), pp. 893–902.
- [13] B. Korte, J. Vygen, Combinatorial Optimization: Theory and Algorithms, *Algorithms and Combinatorics*, 21, Springer-Verlag, 2008, pp. 120–121.
- [14] J. Lee, A First Course in Combinatorial Optimization, *Cambridge Texts in Applied Mathematics*, Cambridge University Press, 2004.
- [15] G.-H. Lin, A. P. Thurber, G. Xue, The 1-Steiner Tree Problem in Lambda-3 Geometry Plane, *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems*, 6 (July 1999), pp. 125–128.
- [16] R. F. Love, G. O. Wesolowsky, S. A. Kraemer, A Multifacility Minimax Location Method for Euclidean Distances, *International Journal of Production Research*, 11 (1973), pp. 37–45.
- [17] C. Monma, S. Suri, Transitions in Geometric Minimum Spanning Trees, *Discrete and Computational Geometry*, 8 (1992), pp. 265–293.
- [18] J. H. Rubinstein, D. A. Thomas, J. F. Weng, Degree-Five Steiner Points Cannot Reduce Network Costs for Planar Sets, *Networks*, 22 (1992), pp. 531–537.
- [19] M. Sarrafzadeh, C.K. Wong, Bottleneck Steiner trees in the plane, *IEEE Trans. Comput.*, 41 (1992), pp. 370–374.
- [20] M. I. Shamos and D. Hoey, Closest-point problems, *Proceedings of the 16-th Annual Symposium on Foundations of Computer Science*, (1975), pp. 151–162.